Introduction to encryption with DES

DES, secret-key decryption

On 15 May 1973, the **NBS** (*National Bureau of Standards*, now called *NIST - National Institute of Standards and Technology*) published a request in the *Federal Register* for an encryption algorithm that would meet the following criteria:

- have a high security level related to a small key used for encryption and decryption
- be easily understood
- not depend on the algorithm's confidentiality
- be adaptable and economical
- be efficient and exportable

In late 1974, IBM proposed "Lucifer", which, thanks to the NSA (National Security Agency), was modified on 23 November 1976 to become the **DES** (*Data Encryption Standard*). The DES was approved by the NBS in 1978. The DES was standardized by the *ANSI* (*American National Standard Institute*) under the name of *ANSI X3.92*, better known as *DEA* (*Data Encryption Algorithm*).

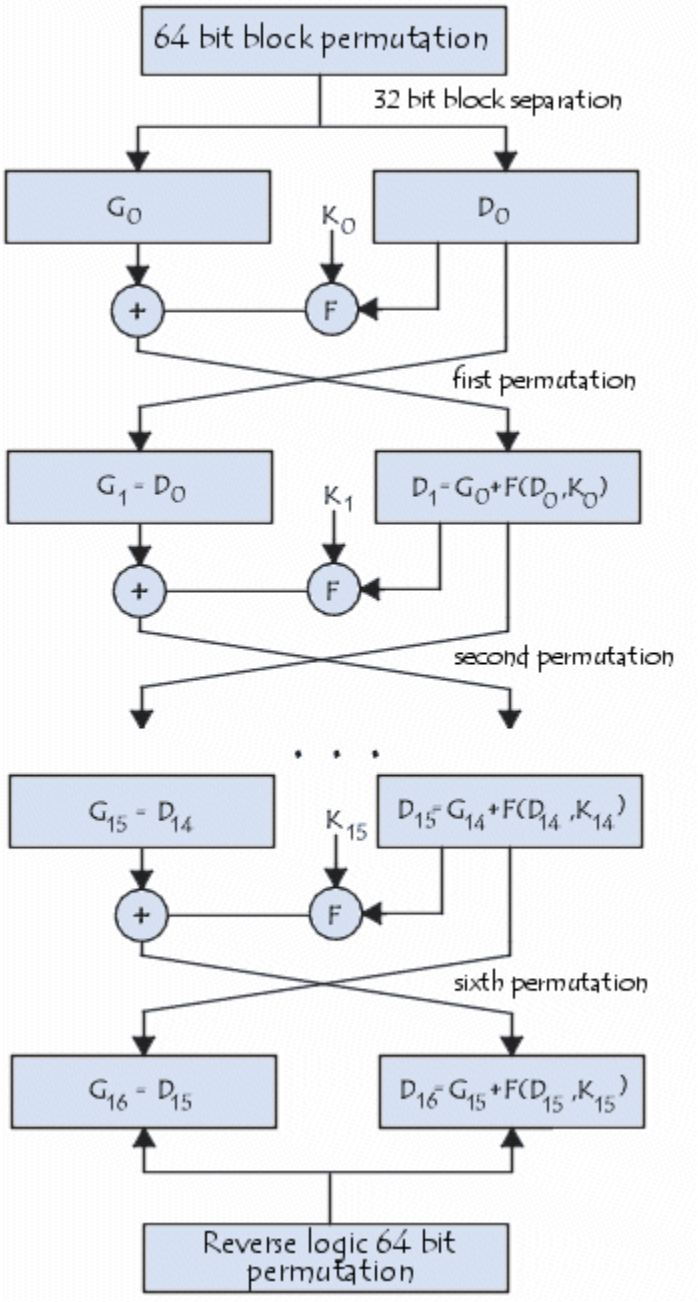
Principle of the DES

It is a symmetric encryption system that uses 64-bit blocks, 8 bits (one octet) of which are used for parity checks (to verify the key's integrity). Each of the key's parity bits (1 every 8 bits) is used to check one of the key's octets by odd parity, that is, each of the parity bits is adjusted to have an odd number of '1's in the octet it belongs to. The key therefore has a "useful" length of 56 bits, which means that only 56 bits are actually used in the algorithm.
The algorithm involves carrying out combinations, substitutions and permutations between the text to be encrypted and the key, while making sure the operations can be performed in both directions (for decryption). The combination of substitutions and permutations is called a **product cipher**.
The key is ciphered on 64 bits and made of 16 blocks of 4 bits, generally denoted k_1 to k_{16}. Given that "only" 56 bits are actually used for encrypting, there can be 2^{56} (or $7.2*10^{16}$) different keys!

The DES algorithm

The main parts of the algorithm are as follows:

- Fractioning of the text into 64-bit (8 octet) blocks;
- Initial permutation of blocks;
- Breakdown of the blocks into two parts: left and right, named *L* and *R*;
- Permutation and substitution steps repeated 16 times (called **rounds**);
- Re-joining of the left and right parts then inverse initial permutation.



Fractioning of the text

Initial permutation

Firstly, each bit of a block is subject to initial permutation, which can be represented by the following initial permutation (*IP*) table:

IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

This permutation table shows, when reading the table from left to right then from top to bottom, that the 58th bit of the 64-bit block is in first position, the 50th in second position and so forth.

Division into 32-bit blocks

Once the initial permutation is completed, the 64-bit block is divided into two 32-bit blocks, respectively denoted **L** and **R** (for left and right). The initial status of these two blocks is denoted $\mathbf{L}_0$ and $\mathbf{R}_0$:

$\mathbf{L}_0$

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8

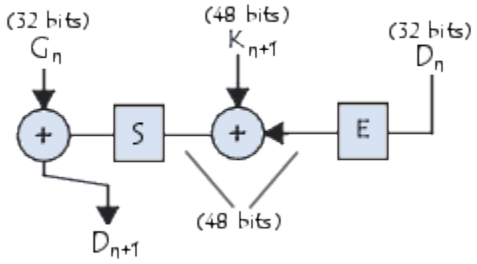
$\mathbf{R}_0$

57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

It is interesting to note that L_0 contains all bits having an even position in the initial message, whereas R_0 contains bits with an odd position.

Rounds

The L_n and R_n blocks are subject to a set of repeated transformations called *rounds*, shown in this diagram, and the details of which are given below:



Expansion function

The 32 bits of the R_0 block are expanded to 48 bits thanks to a table called an *expansion table* (denoted **E**), in which the 48 bits are mixed together and 16 of them are duplicated:

E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

As such, the last bit of R_0 (that is, the 7^{th} bit of the original block) becomes the first, the first becomes the second, etc.
In addition, the bits 1,4,5,8,9,12,13,16,17,20,21,24,25,28 and 29 of R_0(respectively 57, 33, 25, l, 59, 35, 27, 3, 6l, 37, 29, 5, 63, 39, 31 and 7 of the original block) are duplicated and scattered in the table.

exclusive OR with the key

The resulting 48-bit table is called R'_0 or $E[R_0]$. The DES algorithm then *exclusive ORs* the first key K_1 with $E[R_0]$. The result of this *exclusive OR* is a 48-bit table we will call R_0 out of convenience (it is not the starting R_0!).

Substitution function

R_0 is then divided into 8 6-bit blocks, denoted R_{0i}. Each of these blocks is processed by **selection functions** (sometimes called *substitution boxes* or *compression functions*), generally denoted S_i.

The first and last bits of each R_{0i} determine (in binary value) the line of the selection function; the other bits (respectively 2, 3, 4 and 5) determine the column. As the selection of the line is based on two bits, there are 4 possibilities (0,1,2,3). As the selection of the column is based on 4 bits, there are 16 possibilities (0 to 15). Thanks to this information, the selection function "selects" a ciphered value of 4 bits.

Here is the first substitution function, represented by a 4-by-16 table:

S_1

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Let R_{01} equal *101110*. The first and last bits give *10*, that is, 2 in binary value. The bits 2,3,4 and 5 give *0111*, or 7 in binary value. The result of the selection function is therefore the value located on line no. 2, in column no. 7. It is the value *11*, or *111* binary.

Each of the 8 6-bit blocks is passed through the corresponding selection function, which gives an output of 8 values with 4 bits each. Here are the other selection functions:

S_2

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5

2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**

0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
1	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
1	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Each 6-bit block is therefore substituted in a 4-bit block. These bits are combined to form a 32-bit block.

Permutation

The obtained 32-bit block is then subject to a permutation **P** here is the table:

P

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Exclusive OR

All of these results output from **P** are subject to an *Exclusive OR* with the starting$\mathbf{L}_0$ (as shown on the first diagram) to give R1, whereas the initial $\mathbf{R}_0$ gives $\mathbf{L}_1$.

Iteration

All of the previous steps (*rounds*) are repeated 16 times.

Inverse initial permutation

At the end of the iterations, the two blocks L_{16} and R_{16} are re-joined, then subject to inverse initial permutation:

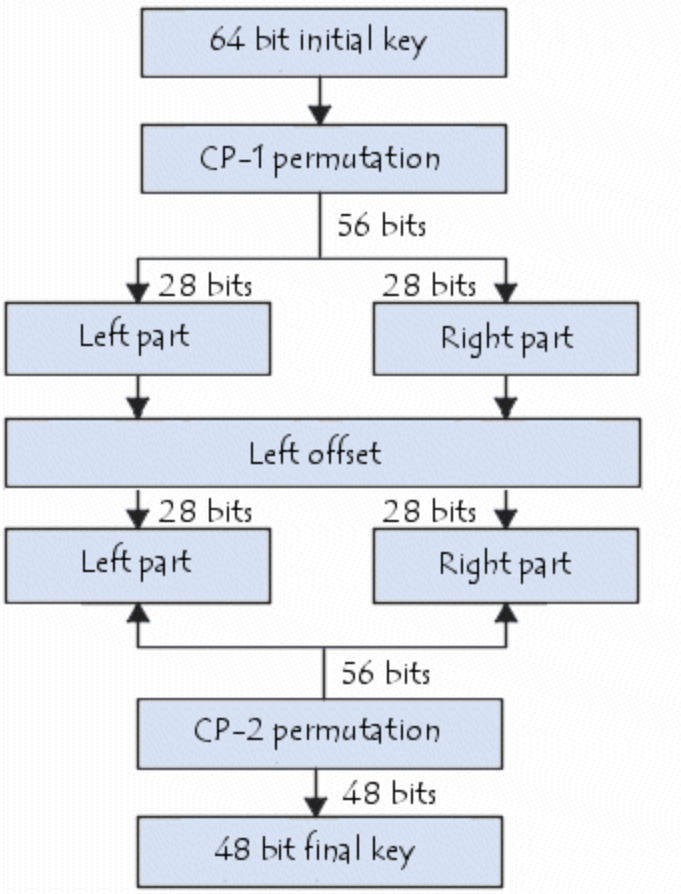
IP-1								
	40	8	48	16	56	24	64	32
	39	7	47	15	55	23	63	31
	38	6	46	14	54	22	62	30
	37	5	45	13	53	21	61	29
	36	4	44	12	52	20	60	28
	35	3	43	11	51	19	59	27
	34	2	42	10	50	18	58	26
	33	1	41	9	49	17	57	25

The output result is a 64-bit ciphertext!

Generation of keys

Given that the DES algorithm presented above is public, security is based on the complexity of encryption keys.

The algorithm below shows how to obtain, from a 64-bit key (made of any 64 alphanumeric characters), 8 different 48-bit keys each used in the DES algorithm:



Firstly, the key's parity bits are eliminated so as to obtain a key with a useful length of 56 bits.

The first step is a permutation denoted **PC-1** whose table is presented below:

PC-1														
	57	49	41	33	25	17	9	1	58	50	42	34	26	18
	10	2	59	51	43	35	27	19	11	3	60	52	44	36
	63	55	47	39	31	23	15	7	62	54	46	38	30	22
	14	6	61	53	45	37	29	21	13	5	28	20	12	4

This table may be written in the form of two tables L_i and R_i (for left and right) each made of 28 bits:

L_i							
	57	49	41	33	25	17	9
	1	58	50	42	34	26	18
	10	2	59	51	43	35	27

	19	11	3	60	52	44	36
R_i	63	55	47	39	31	23	15
	7	62	54	46	38	30	22
	14	6	61	53	45	37	29
	21	13	5	28	20	12	4

The result of this first permutation is denoted L_0 and R_0.
These two blocks are then rotated to the left, such that the bits in second position take the first position, those in third position take the second, etc. The bits in first position move to last position.
The 2 28-bit blocks are then grouped into one 56-bit block. This passes through a permutation, denoted **PC-2**, giving a 48-bit block as output, representing the key K_i.

PC-2	14	17	11	24	1	5	3	28	15	6	21	10
	23	19	12	4	26	8	16	7	27	20	13	2
	41	52	31	37	47	55	30	40	51	45	33	48
	44	49	39	56	34	53	46	42	50	36	29	32

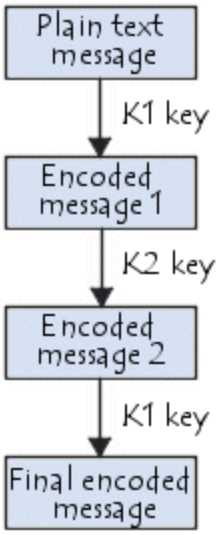
Repeating the algorithm makes it possible to give the 16 keys K_1 to K_{16} used in the DES algorithm.

LS	1	2	4	6	8	10	12	14	15	17	19	21	23	25	27	28

TDES, an alternative to the DES

In 1990, Eli Biham and Adi Shamir developed differential cryptanalysis, which searches for plaintext pairs and ciphertext pairs. This method works with up to 15 rounds, while 16 rounds are present in the algorithm presented above.

Moreover, while a 56-bit key gives an enormous amount of possibilities, many processors can compute more than 10^6 keys per second; as a result, when they are used at the same time on a very large number of machines, it is possible for a large body (a State for example) to find the right key...
A short-term solution involves catenating three DES encryptions using two 56-bit keys (which equals one 112-bit key). This process is called **Triple DES**, denoted *TDES* (sometimes *3DES* or *3-DES*).



TDES is much more secure than DES, but it has the major disadvantage of also requiring more resources for encryption and decryption.
Several types of triple DES encryption are generally recognized:

- DES-EEE3: 3 DES encryptions with 3 different keys;
- DES-EDE3: a different key for each of the 3 DES operations (encryption, decryption, encryption);
- DES-EEE2 and DES-EDE2: a different key for the second operation (decryption).

In 1997, *NIST* launched a new call for projects to develop the **AES** (*Advanced Encryption Standard*), an encryption algorithm intended to replace the *DES*.
The *DES* encryption system was updated every 5 years. In 2000, during the most recent revision, after an evaluation process that lasted for 3 years, the algorithm that was jointly designed by two Belgian candiates, Sirs *Vincent Rijmen* and *Joan Daemen* was chosen as the new standard by *NIST*. This new algorithm, named **RIJNDAEL** by its inventors, will replace the DES from now on.

Source: http://en.kioskea.net/contents/134-introduction-to-encryption-with-des